Reconfigurable Flight Control Design using a Robust Servo LQR and Radial Basis Function Neural Networks

John J. Burken     NASA Dryden Flight Research Center

IJCNN 2005
International Joint Conference on Neural Networks
July 31 – August 5, 2005

ICCNN, Montreal Canada
August 2005
Reconfiguration

Presentation Outline

- Purpose
- Background
- Design Methods Used for Paper
  - Background on Model Reference Adaptive Control (MRAC)
  - Background on Robust Servomechanism LQR
  - Radial Basis Function Neural Networks
- Control Failure Survivability Results
- Results / Time Histories
- Conclusions
  - Remarks
  - Lessons Learned
Control Reconfiguration

General Background / Concepts

- Purpose of Reconfigurable Control / Why?
  - Handle Failures & Land Safely
  - Continue on with Mission
  - Buy More Time to Terminate Flight at a Better Location (UAV)

- Overall Controller Objective.
  - Maintain consistent stable performance in the presence uncertainties and unmodeled dynamics.
Control Reconfiguration

General Background / Concepts

● Why Adaptive Control.
  ➢ Handles Uncertainties and unpredicted parameter deviations.
  ➢ Adaptive control is better than Robust Control w.r.t. slow varying parameters.

● Why Robust Control (Such as Robust LQR servo design)
  ➢ Handles fast varying parameters and unmodeled dynamics.
  ➢ Has good flight experience.

● Solution to Adaptive & Robust control issues.
  ➢ Merge Adaptive augmentation into a Robust Baseline Controller.
• Motivation / Problem Statement  {The Big Picture}
  • Land a damaged airplane or, return to a safe ejection site.
  • Or continue with mission

• General Goals & Objectives
  • Flight evaluation of neural net software.
  • Increased survivability in the presence of failures or aircraft damage.
    • Increase your boundary of a flyable airplane.
    • Increase your chances to see another day.
    • Increase your chances to continue the mission.
Motivation, cont

- Airplanes in the Past Have Landed with Major Failures.

- But possibly not as many safe landings as could have, with adaptive control methods.

- Our Goal is to Increase the Survivability Region for the Pilot without luck or high skill levels or when the pilot is injured.
How do we Reconfigure the Controller (called H or K)

Many ways to adapt to a failure or unknown Plant (G) parameters:
- Adaptation Methods:
  - Non-Learning Methods:
    - Robust Reconfiguration Methods.
    - Fault detection & isolation.
    - Use of smart actuators (Handles only B matrix failures).
    - Reconfigurable Retrofit Architecture methods.
  - Learning Methods:
    - Use of Neural networks
    - To many to list (such as RBF Radial Basis Function)
General Statements on Adaptive Controller

• Two Types of Adaptive controllers
  1. Direct Adaptive
  2. Indirect Adaptive

• The Direct Adaptive Controller Works on the Errors.
  • Needs a Reference Model to Generate $P_{err} = (P_{cmd} - P_{sensor})$
  • The Neural Network “Directly” Adapts to $P_{err}$.
  • Does not need to know the source of error.
    • No Aero Parameter Estimation Needed
    • No need for persistently exciting signals

• The Indirect Adaptive Works on Identifying the source of Error.
  • Does Not Need a Reference Model.
  • Needs to Identify the Aerodynamics that have changed! (PID)
    • PID is Time Consuming and may not be correct.
    • Needs persistently exciting inputs.
Model Reference Adaptive Control (MRAC)

- **Plant**: Actual Plant parameters (G) are unknown.
- **Reference Model**: Ideal response (ym) to cmd r (Use a Stable Reference Model).
- **Adaptation Law**: Is used to adjust controller (H): can be NNs.
Servomechanism Design Methodology

Consider a MIMO system
\[ \dot{X} = Ax + Bu + Ew \quad \text{where} \ x \in \mathbb{R}^n, \ u \in \mathbb{R}^m, y \in \mathbb{R}^p \]
\[ Y = Cx + Du + Fw \]
\[ w = \text{the disturbance (failed surface)} \]
The dynamic controller is
\[ \dot{x}_c = A_c x_c + B_c (r - y) \]
The open loop augmented system is
\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_c
\end{bmatrix} =
\begin{bmatrix}
A & 0 \\
-B_c C & A_c
\end{bmatrix}
\begin{bmatrix}
x \\
x_c
\end{bmatrix} +
\begin{bmatrix}
B \\
-B_c D
\end{bmatrix} U
\]
Suppose the following condition is satisfied
\[ \text{rank}\begin{bmatrix}
\bar{e} & l & A & B \\
-C & D
\end{bmatrix} = n + p \]
The system is controllable and there exist a control law
\[ u = kx + k_c x_c \]

Note:
① LQR Servo = LQR PI
② Jammed or failed surface is treated as a disturbance to the system.
③ Approach is simple to implement.

If this statement is true there exist a closed-loop system that is stable.
Remarks:

- For any such control law, asymptotic tracking and disturbance rejection are achieved; that is, the error goes to zero.
- If the augmented system is controllable, the control law can be conveniently found by applying the linear quadratic regulator (LQR) approach to the augmented system.
- After setting up the augmentation we now need to solve for the gain \((k, k_c)\)
  
  ➤ Just use LQR.
  ➤ This setup allows for a LQR tracker solution.

Control Law

\[ u = kx + k_c x_c \]

\[ e = r - y \rightarrow 0 \]

The augmented system is

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_c
\end{bmatrix} =
\begin{bmatrix}
A & 0 \\
-B_c C & A_c
\end{bmatrix}
\begin{bmatrix}
x \\
x_c
\end{bmatrix}
+ \begin{bmatrix}
B \\
-B_c D
\end{bmatrix} U
\]
- Optimize the following cost function.
  Optimal linear-quadratic-regulator (LQR) problem.

\[ J = \int_0^T (x'Qx + u'Ru)dt \]

- The algebraic Riccati equation

\[ 0 = A'P + PA + Q - PBR^{-1}B'P \]

- And the optimal control is given by:

\[ u(t) = -R^{-1}B'Px(t) = Kx(t) \]
Why Neural Networks?

– Neural Networks are Universal Approximators.
– Minimizes a $H^2$ norm.
– They permit a nonlinear parameterization of uncertainty.
– Why Radial Basis Functions (RBF):
  – RBFs will de-activate when signal is outside “neighborhood”.

Activation function

\[
\phi(x) = e^{-\frac{||x - r||^2}{2\sigma}}
\]
The output of a RBF network with K neurons:

- \( \phi_k(x) \) is the response of the \( k \)th hidden neuron for input vector \( x \).
- \( w_k \) is the connecting weight of the output neuron.

\[
f(x) = NN(x) = \sum_{k=1}^{K} w_k \phi_k(x) + b
\]
Neurons
1 Hidden layer with 4 Neurons and 2 Inputs

\[ f_j = w_0 b + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_1 x_2 \]

\( \phi \) means activation function
2 groups of failures are “common” among aircraft mishaps/crashes.

- Aerodynamic Failures or uncertainties (A Matrix problems / lost aero surfaces, bent wings)
  - Or Not well known aero terms due to modelling errors.

- Control Failures (B Matrix problems / jammed control surfaces)
  - Right stab jammed at 8. deg from trim
Control Reconfiguration Results

- Time History of Surface Failure (B matrix)
- Failure = Right Stabilator Jammed.
  - At time = 10 seconds / 8 deg from trim.
  - At time = 30 seconds Failure goes away (crew fixed the failure).
- Neural Networks
  - Neural Networks turned off for the first run.
  - Neural Networks turned on for second run.
  - Without Dead Zones.
Robust Model Reference Adaptive Control Design

F-18 LQR-Tracker (Robust Servo LQR)
Model Reference Adaptive Control System Design

March 01, 2005
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets
Lat/Dir Axis Data

Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets
Control Reconfiguration Results

- Time History of Surface Failure (B matrix)
- Failure = Right Stabilator Jammed.
  - At time = 10 seconds / 8 deg from trim.
  - At time = 30 seconds Failure goes away (crew fixed the failure).
- Neural Networks
  - Neural Networks turned off for the first run.
  - Neural Networks turned on for second run.
  - With Dead Zones & 20% decrease in learning rates.
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets

NN with Dead-Zones & Slower Learning
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets

NN with Dead-Zones & Slower Learning
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets

NN with
Dead-Zones & Slower Learning
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets

Neural Network Signals

NN with Dead-Zones & Slower Learning
Failure = Right Stab 8. deg at 10 seconds with & without NN
Failure goes away at 30 seconds / Pilot Input is Roll doublets

NN with Dead-Zones & Slower Learning
Conclusions & Remarks

- Method presented:
  - Robust LQR Servomechanism design with Model Reference Adaptive Control
    - Reference Model was a “health” aircraft.
  - Used Radial Basis Function Neural Networks

- Results:
  - LQR Servomechanism behaved well with a failure.
  - Using the Neural Networks improved the tracking compared to not using the neural networks.

- Lesson learned:
  - Test the removal of the failure with Neural Networks active to ensure good performance.
    - The crew could fix the problems and you don’t want the adaptive system to go unstable.
Appreciations / Thanks

- Eugene Lavretsky, Ph.D  Boeing
- Ping Lu,  Iowa State University